REMARKS

Not only does the cited reference fail to teach using different instruction sets, for each of three processors, but it fails to suggest using three different types of processors (as argued in response to the last office action).

The language "The main reason why two different types of processors are necessary is because of the level of processing" is cited. See column 14, lines 47-49. How this is transformed into using more than two types of processors is hard to understand. A suggestion that it teaches using "at least" two processors is unsupported by the reference to the extent it is intended to assert that the reference teaches using more than two different processors. The reference says you must use two different types of processors. It does not suggest that you must use three different types of processors. It does not suggest that you must use "at least" two different types of processors. Instead, it is explicit and limiting that you must use two different types of processors. Therefore, the arguments made in paragraph 24 of the office action are not supported by the language of the reference.

Moreover, nothing in the reference suggests using different instruction sets, even for the two different processors. Just because the processors are different does not mean that their instruction sets are different. The reference fails to teach the point claimed and the deductions relied upon are unsupported by the reference. And even if the reference taught using two different instruction sets, it does not teach using different instruction sets for the specific processors set forth in the claim.

The suggestion that an instruction set for a processor corresponds to the inputs that are received is baseless. An instruction set is a well known term of art and the attempt to redefine it without any support is improper. Namely, the apparent attempt to redefine "inputs" to be "instruction sets" is unduly strained. Processors receive inputs and these are not instruction sets. Moreover, the language of the claim precludes the strained interpretation propounded. The claim requires that each of the processors has a different instruction set than all the other processors. Thus, the claim requires that they have instructions, not that they receive instructions.

For example, as set forth in the attached material from the Computer Desktop Encyclopedia, an instruction set is "The repertoire of machine language instructions that a computer can follow (from a handful to several hundred). It is a major architectural component

and is either built into the CPU or into microcode. Instructions are generally from one to four bytes long."

Nothing in the materials cited at column 11, line 55 through column 12, line 12 suggest anything about the instruction set of the so-called transfer processor 11. The material at column 11, line 65 that refers to instructions has nothing to do with the instruction set and has nothing to do with inputs. Similarly, the instruction streams referred to in column 12, lines 8 and 9 are streams of instructions that come through the transfer processor so that it can transfer instructions. It transfers instructions, presumably using an instruction set which is nowhere discussed in any of the cited materials. The instruction streams are not the instruction set of the transfer processor. The problem is that the instructions that are transferred are not the instruction set of the processor. The instruction set of the processor are those instructions stored within the processor that allows it to transfer the instruction streams. The instruction streams are simply the data that is received and transferred. How it does what it does is dependent on its instruction set. Its instruction set is nowhere described, discussed, or in any way explained.

In effect, the rejection is based on nothing within the reference. There is no basis whatsoever to conclude that the instruction set of any processor within the reference is different from any other.

Moreover, even if two types of processors with two different types of instruction sets were used, there is no suggestion that the specific processors claimed would have three different instruction sets.

Therefore, reconsideration is requested.

Respectfully submitted,

Date: April 2, 2007

Timothy N. Trop, Reg. No. 28,994
TROP, PRUNER & HU, P.C.
1616 South Voss Road, Suite 750
Houston, TX 77057-2631
713/468-8880 [Phone]
713/468-8883 [Fax]
Attorneys for Intel Corporation

Computer Desktop Encyclopedia

Ninth Edition

Alan Freedman

Osborne/McGraw-Hill
New York Chicago San Francisco
Lisbon London Madrid Mexico City Milan
New Delhi San Juan Seoul Singapore Sydney Toronto

instance variable In object-oriented programming, a variable used by an instance of a class. It holds data for a particular object. Contrast with *class variable*. See *class*.

instantiate In object technology, to create an object of a specific class. See *instance*.

instant messaging A computer conference using the keyboard (a keyboard chat) over the Internet between two or more people. Instant messaging is not a dial-up system like the telephone; it requires that both parties be online at the same time. You have to put the names of people you want to instant message with in a list, and when any of those individuals log on, you are "instantly" notified so that you can begin an interactive chat session. AOL's Instant Messenger (AIM), Microsoft Network Messenger Service (MSNMS), ICQ and Yahoo! Messenger are the major instant messaging services.

In the business world, instant messaging is often used to avoid telephone tag, or to find out if a person is available to take a phone call. Many instant messaging sessions wind up as traditional telephone calls. However, instant messaging is expected to be the catalyst for IP-based phone calls initiated directly from the computer to provide a seamless move from typing to talking. See *IMUnified* and *Jabber*.

instant messenger The software that provides instant messaging services. See *instant messaging* and *AIM*.

instant print The ability to use the computer as a typewriter. Each keystroke is transferred to the printer.

instant replay See *PVR*.

Institute for Certification See *ICCP*.

instruction (1) A statement in a programming language.
(2) A machine instruction.

instruction cycle The time in which a single instruction is fetched from memory, decoded and executed. The first half of the cycle transfers the instruction from memory to the instruction register and decodes it. The second half executes the instruction.

instruction mix The blend of instruction types in a program. It often refers to writing generalized benchmarks, which requires that the amount of I/O versus processing versus math instructions, etc., reflects the type of application the benchmark is written for.

instruction register A high-speed circuit that holds an instruction for decoding and execution.

instruction repertoire Same as *instruction set*.

instruction set The repertoire of machine language instructions that a computer can follow (from a handful to several hundred). It is a major architectural component and is either built into the CPU or into microcode. Instructions are generally from one to four bytes long.

instruction time The time in which an instruction is fetched from memory and stored in the instruction register. It is the first half of the instruction cycle.

insulator A material that does not conduct electricity. Contrast with *conductor*.

Int A programming statement that specifies an interrupt or that declares an integer variable. See *interrupt* and *integer*.

Int 13 A DOS interrupt used to activate disk functions, such as seek, read, write and format.

Int 14 A DOS interrupt used to activate functions on the serial port (COM1, COM2, etc.). See *NASI*.